REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
10-04-2015	Final Report	5-Dec-2011 - 4-Jun-2013

4. TITLE AND SUBTITLE
Final Report: Scalable Low-Power Deep Machine Learning with Analog Computation

5a. CONTRACT NUMBER
W911NF-12-1-0017

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
411359

6. AUTHORS
Jeremy Holleman, Itamar Arel, Jennifer Hasler

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
University of Tennessee at Knoxville
Office of Sponsored Programs
1534 White Avenue
Knoxville, TN 37996 -1529

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
61325-EL-IRP.4

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
Over the course of this project we have accomplished the primary objective of this seedling project to develop analog computational circuits for deep machine learning. We have performed extensive simulations investigating the magnitude of errors expected from analog computational elements and their impact on the overall learning performance. We have also fabricated and tested a prototype chip, which demonstrates successful execution of an unsupervised machine learning task. We have further designed a second more complex learning chip. We have made significant innovations to the algorithm to facilitate analog implementation. We have also implemented

15. SUBJECT TERMS
Analog Machine Learning, analog circuits, deep machine learning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Jeremy Holleman
UU	UU	UU	UU		19b. TELEPHONE NUMBER: 865-974-5442

Report Title

Final Report: Scalable Low-Power Deep Machine Learning with Analog Computation

ABSTRACT

Over the course of this project we have accomplished the primary objective of this seedling project to develop analog computational circuits for deep machine learning. We have performed extensive simulations investigating the magnitude of errors expected from analog computational elements and their impact on the overall learning performance. We have also fabricated and tested a prototype chip, which demonstrates successful execution of an unsupervised machine learning task. We have further designed a second more complex learning chip. We have made significant innovations to the algorithm to facilitate analog implementation. We have also implemented several adaptive circuits in a field-programmable analog array (FPAA).

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received		Paper
04/10/2015	1.00	Steven Young, Junjie Lu, Jeremy Holleman, Itamar Arel. On the Impact of Approximate Computation in an Analog DeSTIN Architecture, IEEE Transactions on Neural Networks and Learning Systems, (05 2014): 934. doi: 10.1109/TNNLS.2013.2283730
TOTAL:	**1**	

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received		Paper
TOTAL:		

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper	
04/10/2015	2.00	Junjie Lu, Steven Young, Itamar Arel, Jeremy Holleman. An analog online clustering circuit in 130nm CMOS, 2013 IEEE Asian Solid-State Circuits Conference (A-SSCC). 11-NOV-13, Singapore, Singapore. : ,
04/10/2015	3.00	Junjie Lu, Jeremy Holleman. A floating-gate analog memory with bidirectional sigmoid updates in a standard digital process, 2013 IEEE International Symposium on Circuits and Systems (ISCAS). 19-MAY-13, Beijing. : ,

TOTAL: **2**

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received	Paper

TOTAL:

Number of Manuscripts:

Books

Received Book

TOTAL:

Received Book Chapter

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED	Discipline
Junjie Lu	0.94	
Alexander Saites	0.50	
FTE Equivalent:	**1.44**	
Total Number:	**2**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
Jeremy Holleman	0.11	No
Itamar Arel	0.08	No
FTE Equivalent:	**0.19**	
Total Number:	**2**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period: 1.00

The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:..... 1.00

The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:..... 1.00

Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):..... 1.00

Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:..... 0.00

The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense 0.00

The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:..... 1.00

Names of Personnel receiving masters degrees

NAME
Total Number:

Names of personnel receiving PHDs

NAME
Total Number:

Names of other research staff

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Sub Contractors (DD882)

Inventions (DD882)

Scientific Progress

See attachment

Technology Transfer

None

Final Report

Proposal Title: Scalable Low-Power Deep Machine Learning with Analog Computation
Authors of Report: Drs. Jeremy Holleman (PI, UTK), Itamar Arel (UTK), Jennifer Hasler (GT)
Contract Number: W911NF-12-1-0017

Performing Organizations:

The University of Tennessee
Dept. of Electrical Engineering & Computer Science
1520 Middle Dr.
Knoxville, TN 37996-2250

Georgia Institute of Technology
School of Electrical and Computer Engineering
777 Atlantic Drive NW
Atlanta, GA 30332-0250

Technical Point of Contact: Dr. Jeremy Holleman (jhollema@utk.edu)
Date: July 19, 2013 **khkjh**
Reporting Period: December 5, 2011 – June 4, 2013

Abstract: Over the course of this project we have accomplished the primary objective of this seedling project to develop analog computational circuits for deep machine learning. We have performed extensive simulations investigating the magnitude of errors expected fromby analog computational elements and their impact on the overall learning performance. We have also fabricated and tested a prototype chip, which demonstrates successful execution of an unsupervised machine learning task. We have further designed and fabricated a second more complex learning chip. We have made significant innovations to the algorithm to facilitate analog implementation. We have also implemented several adaptive circuits in a field-programmable analog array (FPAA).

Supported Students & Personnel

- **Junjie Lu:** Graduate student, 100% supported at 25% FTE (June 2012 – July 2012) 50% FTE (August 2012 – present)
- **Jeremy Holleman:** Faculy, 95% FTE for two months (May, June 2012)
- **Itamar Arel:** Faculty, 16.7% FTE for nine months
- **Alexander Saites:** Approximately 40 hours/month, nine months

Scientific Progress & Achievements

Over the course of this project, we have accomplished the primary goals laid out in the proposal. In the following subsections, we will summarize the main achievements.

Task I: Cortical Circuit Design

Within the circuit design task, we have completed the following tasks:

- Designed a novel floating-gate memory cell capable of random-addressable tunneling without a high-voltage switch or cell-local voltage booster
- Designed a current-mode computational cascade for clustering
- Modeled the non-ideal aspects of analog computation to enable system-level evaluation of the impacts of non-ideal analog computation
- Implemented and fabricated a complete 8-input 4-centroid analog clustering circuit
- Tested the prototype clustering circuit, demonstrating its successful operation
- Evaluated the power efficiency of analog and digital computation
- Refined the clustering circuit design
- Implemented a seven-node three layer clustering circuit

In the next several pages, we will describe the circuit advances in detail.

Floating-Gate Analog Memory

Any iterative learning system must store parameters and update them as new data is observed. Storage devices used for this purpose should ideally have the following properties:

- Non-volatile storage or long retention times so that observations can be accumulated over long periods of time.
- Very small minimum update magnitude to allow the appropriate learning rate to be applied
- Symmetric increment and decrement, so that equilibrium is achieved when an equal number of increment and decrement operations are applied.
- Very low power consumption
- Minimal silicon area usage

For the analog computational system developed within this project, parameters are stored using an analog floating-gate (FG) memory cell. The update magnitude is pulse-width controlled, and negative feedback is employed to keep the floating gate voltage constant so as to get a smooth sigmoid update rule necessary for stable adaptation. The novel update scheme allows random-accessible control of both tunneling and injection without high-voltage switches, charge pumps or complex routing. The memory does not require special masks or a second poly-silicon layer, and is fully compatible with modern digital CMOS process.

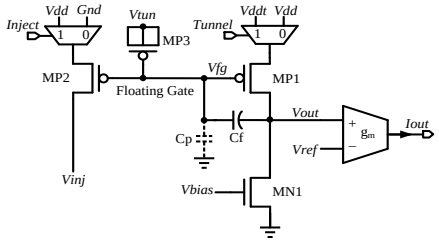


Figure 1: Schematic of the floating-gate analog memory cell.

The schematic of the proposed FG analog memory cell is shown in Figure 1. The gate of MP1-MP3 and the top plate of C_f form the FG. The stored charge can be modified by the injection transistor MP2 and the tunneling transistor MP3. MP1 together with the current source MN1 forms a single-ended inverting amplifier. The amplifier has C_f as its feedback capacitor and the FG voltage V_{fg} as its input. The two MUXs at the sources of MP1 and MP2 control the tunneling and injection of the FG, which will be discussed later.

The transconductor g_m, implemented as a simple operational transconductor amplifier (OTA), converts voltage V_{out} to output current I_{out}. V_{ref} determines the nominal voltage of V_{out} during operation; it is set close to V_{DD} to avoid unwanted injection from MP1.

The negative feedback loop comprising the inverting amplifier and C_f keeps the FG voltage Vfg constant, ensuring an output-independent update of V_{Out}. Tunneling or injection to the FG node changes the charge stored in C_f, and therefore changes the output of the amplifier by $\Delta V_{out} = \Delta Q / C_f$. The loop gain provided by the amplifier attenuates the voltage variation of V_{fg} due to the swing of V_{out}, and suppresses the noise from the ground supply. To have a value-independent update as well as good ground noise rejection, C_f is made much larger than C_p, and MN1/MP1 are made long to increase their drain resistances.

mvsdj ekjt sdfjm qwij

jkhkjh

Floating gate charge modification modeling

The proposed analog memory uses Fowler–Nordheim tunneling to remove the electrons from the FG and decrease output current. The applied electric field across MP3's gate oxide reduces its effective thickness, increasing the probability of electrons' tunneling through it. The tunneling current depends on process-specific parameters as well as the voltage across the oxide. It can be fit to an empirical model presented previously by Rahimi et al. We inferred the tunneling current from the output voltage rate of change in order to fit the model. This effort has allowed us to incorporate measurements into circuit models to guide other designs.

Selective and Value-independent Update Scheme

The proposed tunneling scheme exploits the steep change of tunneling current with regard to Vox to achieve a good isolation between selected and unselected memories. Selectively applying a high voltage to the tunneling oxide of an individual memory cell presents difficulties. Multiplexing a high voltage requires high-voltage switches that are not available in standard CMOS processes. The high voltage can be generated locally to avoid the need for switches, but this method requires a voltage doubler for each memory cell, leading to large silicon area requiremenets. Instead we simply connect the Vtun of all the memory cells to an off-chip voltage and manipulate Vfg to achieve selective tunneling. As Vox = Vtun – Vfg, reducing Vfg is equivalent to increasing Vtun on its effect on the tunneling current. The operation of this scheme can be described by Figure 2, showing the memory cell, but omitting components irrelevant to tunneling process. To show how Vox is changed, typical nodal voltages are annotated. Reducing supply voltage of the selected memory effectively reduces Vfg and increases Vox. In our design, V_{DD} is switched from 3 V to 1 V so that V_{ox} is increased from 4.4 V to 6.4 V. According to the standard Fowler-Nordheim tunneling equations, the tunneling current is 4.8 fA at Vox = 6.4 V, while it's 1.1e−22 A at Vox = 4.4 V, indicating an isolation over 7 orders of magnitude. In practice, the leakage at lower Vox may be degraded by direct tunneling, which is a weaker function of the applied field, and parasitic coupling. Isolation of 83.54 dB is observed in measurement.

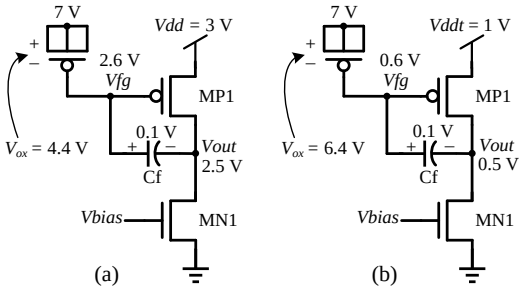


Figure 2: Simplified schematics and typical nodal voltages of memory cells (a) not selected. (b) selected for tunneling.

Injection selectivity is achieved by switching the source voltage of the injection transistor MP2. The source of MP2 in the unselected memory is connected to ground while the one in the selected cell is connected to Vdd, enabling injection. The INJ terminals of all the memory cells are connected to an off-chip voltage, facilitating fine-tuning of injection rate. Is is a constant for all memory cells since it's a mirrored version of the biasing current in MN1. Therefore, the injection is also selective and value-independent.s

Memory Cell Measurement Results

The area of a single memory cell is 35x14 μm^2. It operates at 3 V power supply and consumes 15 nA with an output range of 0-10 nA. The biasing current is tunable and allows the designer to balance between range, speed and power consumption. The performance summary is listed in Table I.

After programming thirty memory cells to values between 1 and 9 nA the standard deviation of the programming error was measured to be 76 pA, indicating a 7-bit programming resolution. This resolution is limited by the noise and resolution of the transimpedance amplifier (TIA) that measures the output current

and the DAQ. The memory output noise is 20.5 pArms over 10 KHz bandwidth from simulation, indicating a 53.8 dB dynamic range. Larger dynamic range and higher resolution can be obtained by increasing the biasing current.

Table 1: Floating-Gate Memory Cell Summary

Parameter	*Value*
Area	35x14 μm^2
Power supply	3 V
Programming resolution	7 bits
Dynamic range	53.8 dB
Programming isolation	86.5 dB

During the same test, the stored values of the other 31 unselected cells are monitored to measure the writing crosstalk. There is no observable injection crosstalk because the drain current can be completely turned off in the unselected cells. Tunneling crosstalk is very small, comparable to the noise floor of the measurement system. By averaging the values among 31 cells, a 471 fA tunneling crosstalk is estimated with a 10 nA writing magnitude in the selected cell, corresponding to an 86.5 dB isolation.

Clustering Circuit

One of the primary accomplishments of this project is the design and fabrication of a complete clustering circuit suitable for use as a single node within a deep learning system. The circuit accepts eight input dimensions and clusters the input vectors to four centroids. The top-level architecture is shown in Figure 1.

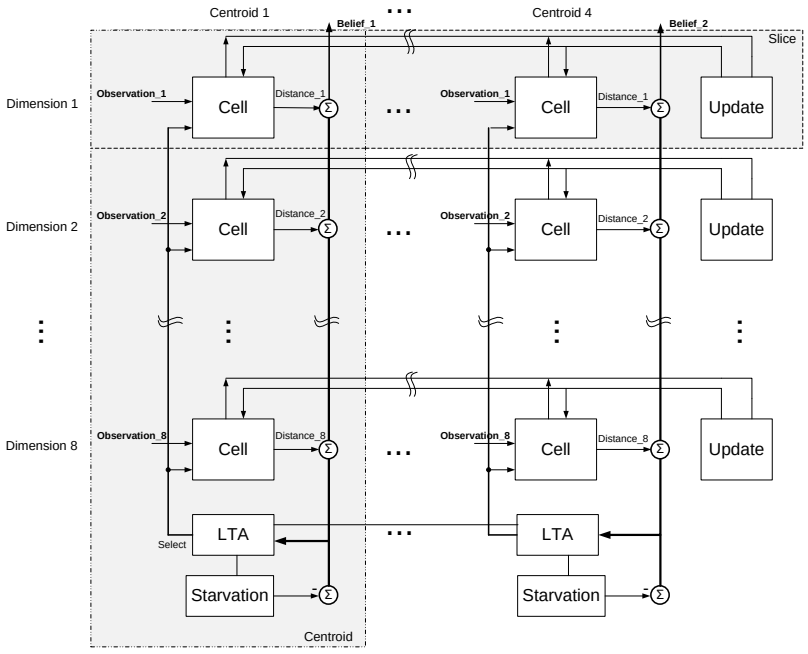


Figure 3: Top-level architecture of the first prototype node circuit

Figure 4(a) illustrates a block diagram of the unit clustering cell, which learns the mean and variance of a subset of its inputs, selected by the *Update* input. The unit cell also outputs a current proportional to either $(\mu\text{-}Observation)^2$ or $(\mu\text{-}Observation)^2/\sigma^2$, which can be summed across dimensions to compute the Euclidean or Mahalanobis distance, respectively. The mean and variance are stored in floating gate memory cells as described above. Figure 4(b) shows a translinear current-mode circuit for computing the distance between an observation and the stored mean. This circuit implements the x^2/y block in Figure 4(a).

Each centroid provides a current-mode representation of its distance from the observation to the loser-take-all (LTA) circuit, described below, which then determines the smallest input, corresponding to the centroid most likely to have generated the input.

In the adaptation phase, the unit outputs both 1-D element of Manhattan distance $d_{ik}^{Man} = |y_{ik}^{\mu} - x_i|$, and Euclidean distance d_{ik}^{Euc}. The mean and variance error signals between the centroid and input vector are propagated to the memory adaptation (MA) circuit,

$$err_{ik}^{\mu} = x_i - y_{ik}^{\mu},$$
$$err_{ik}^{\text{var}} = d_{ik}^{Euc} - y_{ik}^{\text{var}}. \quad (2)$$

Note that the Euclidean distance is reused to construct the error signal for variance because of its simple quadratic form. Then the MA circuit generates update signals UPD_i for the centroid mean and variance memories. Although the UPD is broadcasted, logic in the MDCs ensure that only the ones selected by the LTA are updated. The amount of update is proportional to the error in (2),

$$y_{ik}^{\mu}(n+1) = y_{ik}^{\mu}(n) + \alpha \cdot err_{ik}^{\mu},$$
$$y_{ik}^{\text{var}}(n+1) = y_{ik}^{\text{var}}(n) + \beta \cdot err_{ik}^{\text{var}}, \quad (3)$$

where α and β are the learning rates. From (3), the memory values follow the exponential moving averages and converge to the true mean and variance of the input data clusters. The memory adaptation is fully parallel.

The proposed clustering circuit also constructs confidence scores for each centroid with regard to the input vector based on Mahalanobis distances with a diagonal covariance matrix

$$d_j^{Mah} = \sum_i \frac{(x_i - \mu_{ij})^2}{\sigma_{ij}^2}, \quad (4)$$

where μ_{ij} and σ_{ij}^2 are the estimated mean and variance of the input data cluster. By including variance, the Mahalanobis distance better estimates the probability that a given input came from a centroid.

Through system-level simulations we determined that the use of Mahalanobis distance in winner selection results in a positive feedback effect. When an observation is classified to a relatively distant centroid, that centroid's variance estimate is increased. The increased variance in turn enables the centroid to be assigned increasingly distant observations, further increasing its variance estimate. To avoid this positive feedback

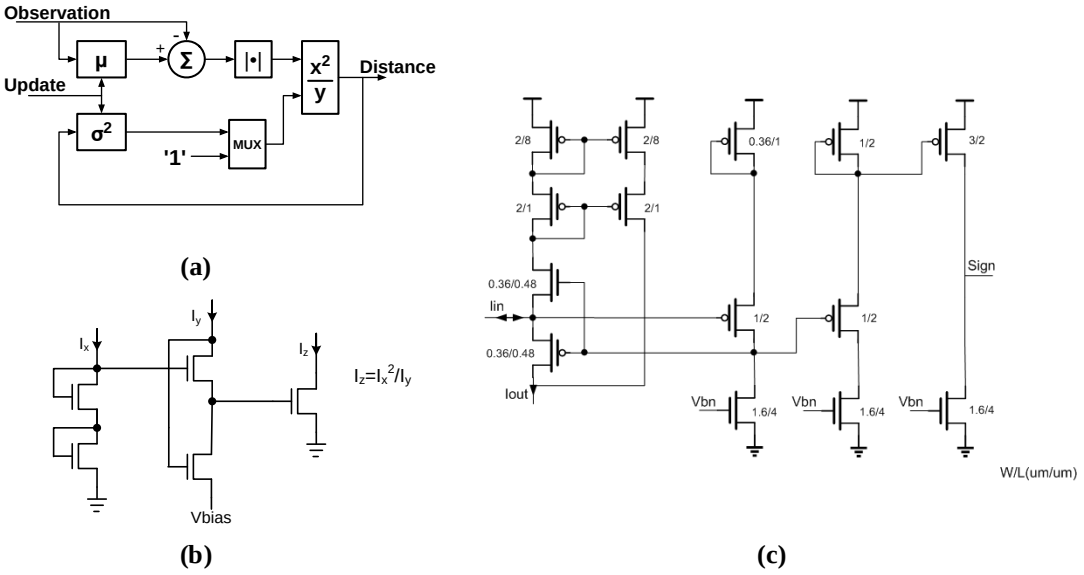


Figure 4: (a) Single unit of the clustering circuit (b) Translinear squaring-dividing cell for calculation of distance and belief. (c) Current-mode absolute value circuit gate memory cell.

and ensure robust behavior, Euclidean distance is used in our classification stage.

A time-domain loser-take-all (LTA) network, described below, common to the columns searches for the single centroid k with minimum distance to $\boldsymbol{x}$ and sets SEL_k to high. The selected column of MDCs is then connected to a memory adaptation (MA) circuit.

To address unfavorable initial condition, a starvation trace algorithm is implemented [12]. Current is injected to decrease the apparent distance of the centroid which has not been selected by the LTA for a long time. As a result, centroids initialized too far away from populated regions of the input space will slowly move toward the input samples.

Time-Domain Loser-Take-All (TD-LTA) Circuit

The LTA circuit receives the Euclidean distances d_j^{Euc}, and search for the centroid with smallest distance. It consists of 4 LTA cells interconnected as shown in Figure 5(a). The LTA cell shown in Figure 5(b) operates in time domain and exploits the dense digital cells in modern process. The typical timing diagram of the "loser" and a "non-loser" cell is plotted in Figure 5(c). The capacitor Cp is initially precharged to *Vdd*, and is discharged by the input current when *Start* goes high (t_0). For the "loser" cell, the threshold crossing of the comparator is the latest among the 4 cells (t_2), so the data input *D* of its D-latch is low when *Latch* goes high. For the "non-loser" cell, *D* is high when *Latch* goes high (t_1). Therefore, the output of the "loser" is latched to low while those of the others latched to high. At the end of LTA phase, *Start* latches all the cells regardless of V_{IN}. Additional logic, omitted from Figure 5 for clarity, prevents the selection of multiple losers.

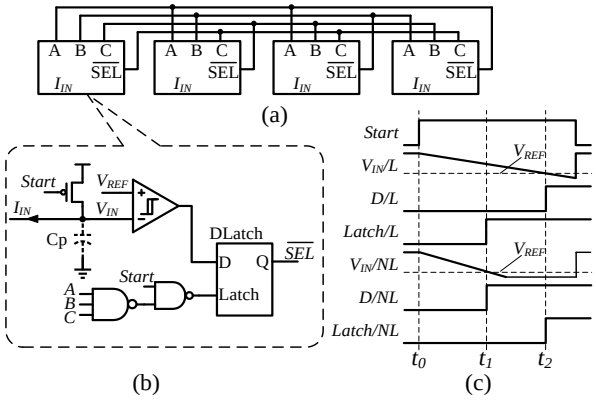


Figure 5: (a)) Array of TD-LTA cells forms a 4-input LTA evaluator (b) Single cell of the time-domain winner-take-all circuit (c) Illustrative waveforms of a loser (L) and non-loser (NL) cell

Compared to continuous time (CT) implementations similar to [11], the proposed TD-LTA can potentially yield lower power-delay product if Cp is realized with the parasitic capacitance at the input node. Since the input current is off before LTA phase, for CT-LTA, the input node has a voltage swing of about V_{th}, and a time constant of Cp/g_m, where V_{th} and g_m are the threshold voltage and transconductance of the input transistor. Neglecting the bias consumption and internal-node settling times, assuming $gm/Id \approx 20$ for transistors biased in weak inversion, and accounting for slewing and 3τ settling time, the current-delay product of CT-LTA is

$$I_{IN} \cdot t_{settle,CT} = (V_{th} + 0.15) \cdot Cp, \quad (6)$$

while the current-delay product of the TD-LTA is

$$I_{IN} \cdot t_{d,TD} = (Vdd - V_{REF}) \cdot Cp. \quad (7)$$

Comparing (6) and (7), with same Cp, the TD-LTA has smaller current-delay product if the V_{REF} is set less than $V_{th}+0.15$ below *Vdd*. In the prototype, V_{REF} is adjustable to balance between energy efficiency and accuracy. The accuracy of the TD-LTA depends on the matching of Cp, and a good matching of wiring capacitances can be achieved with layout regularity.

Clustering Measurement Results

Figure 6 shows a micrograph of the prototype. It has 8 input dimensions and 4 centroids and can be easily scaled up. It consumes 15 µW with 3 V supply. Test signals are provided to and acquired from the chip using PC-based data acquisition hardware and off-chip current-voltage converters on a custom test board.

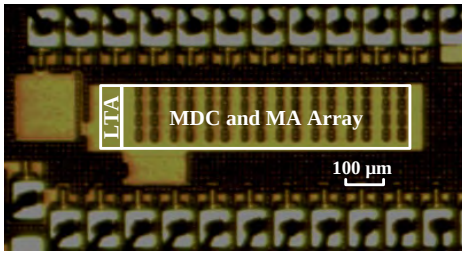


Figure 6: The micrograph of the prototype clustering circuit. In a 130-nm standard CMOS process, it occupies 0.18 mm2 of active area including the programming shift registers and biasing circuits.

To characterize the distance computation block, the stored centroid mean y^{μ} of a MDC cell is programmed to 10 nA, and its input x is swept from 0 to 20 nA. The Mahanolobis distance output with varying centroid variance y^{var} is measured and plotted in Figure 7. The block shows good linearity over the entire input range, and a power law output corresponding to (4).

The classification test was performed by programming the centroids to fixed positions and disabling memory adaptation. The inputs are equally spaced and randomly presented to the circuit. To allow easier visual interpretation, only 2 out of 8 dimensions of input vectors are shown. The results are color-coded and the measured decision boundaries show good matching with the ideal boundaries, illustrated in Figure 8(a). The prototype circuit runs classification at a speed of 16 kHz, limited by the settling time of the input current.

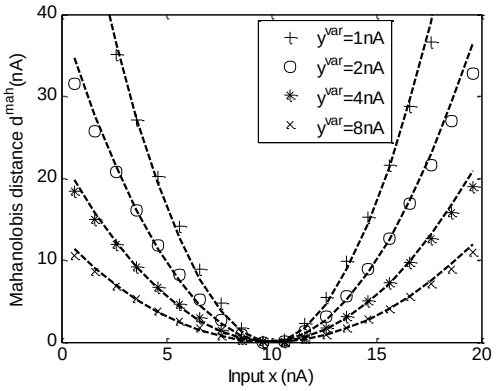
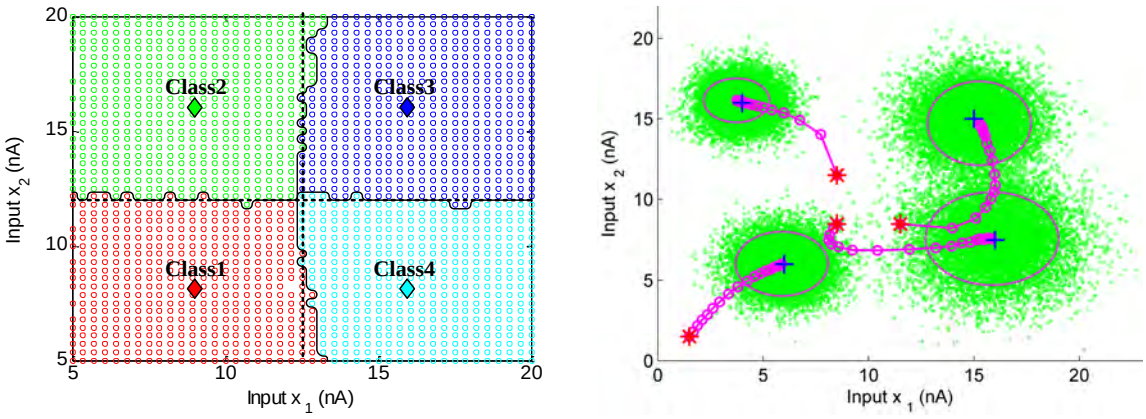


Fig. 7. Transfer characteristics of the D³ block. The markers are the measured data and the dashed lines are curves fitted to the quadratic form $a(x-10)^2$, where a is the fitting parameter.

We demonstrated the full functionality of the prototype by solving a clustering problem. 40000 8-dimentional vectors were generated as the inputs to the circuit. The data set contains 4 underlying clusters, each drawn from Gaussian distributions with different means and variances. Initially the centroids were programmed to different means and equal variance. The initial condition is not critical since the circuit will adaptively adjust to the inputs. The centroid means were read out every 0.5 s during the test, plotted on top of the data scatter in Figure 8(b), and shown together with the learned variance values at the end of test. Again 2 of 8 dimensions are shown. The centroids converge robustly to the clusters' centers of mass despite overlapping clusters. The extracted variances match well with the true variance. The clustering test takes 10 s at a speed of 4 kHz; higher speed is possible at the cost of lower learning rate.



(a) (b)

Figure 8: (a) Classification results. The 4 centroids are shown in diamond shapes. The circuit assigns the input data to different centroids based on the Euclidean distances. The results are color coded and the measured decision boundaries are shown as solid lines and ideal boundaries as dashed lines. (b) Clustering test result. The input data are shown in green circles. The initial position of centroid means are marked with red stars, their positions for every 0.5 second during the test are plotted with magenta circles connect by lines. The extracted variances of the clusters are plotted with dashed ellipses in magenta, and the true means of data clusters are marked with blue crosses.

The measured performance is summarized in Table. II. The bottleneck of the operation speed is the settling time of inputs, because small currents are delivered from off-chip. Simulation results indicate that the internal circuitry is operational at up to 50 kHz. Therefore, the energy efficiency of this proof-of-concept prototype can be largely improved by pre-distorting the input current to voltage or using on-chip D/A converters.

Table II: Performance Summary

Technology	130-nm digital CMOS
Total Area	0.9 × 0.2 mm^2 (8 × 4 array)
MDC Cell Area	90 × 30 μm^2
Power Consumption	15 μW @ 3V
Classification Speed	16 kHz
Clustering Speed	4 kHz

Comparison of Analog and Digital Efficiency

To compare the proposed analog implementation to its digital counterpart, we focus on the translinear operator shown in Figure 4(b), because it is the most computation-intensive block, and will dominate the power consumption when the system is scaled up. The translinear circuit has an SNR of 130 with typical bias current, and operates with an energy efficiency of 76.2 fJ/OP. A digital arithmetic block computing x^2/y with 7-bit resolution was synthesized in the same process and yields an energy efficiency of 1 pJ/OP, suggesting that the analog implementation in this work can have more than one order of magnitude higher energy efficiency. Further advantage over digital realization can be expected if the cost-free current-mode summing, efficient TD-LTA and memory reading/writing are included to the comparison.

Clustering Circuit Improvements

Based on the results from the first prototype, we designed a second, larger-scale, prototype. While the first chip included a single clustering circuit, the second chip includes a three-layer hierarchy of seven nodes arranged in a 4-2-1 pyramid. Each node includes a clustering circuit as well as additional circuitry to interface to adjacent nodes. The layout of the second chip is shown in Figure 9. We have also made several improvements to the clustering circuit since the first version.

- A system model with mismatch error sources was constructed. Based on the error-performance curve from the system simulation, the sizes of most transistors are reduced to just meet the system accuracy requirements. This reduces chip area as well as the energy per operation by reducing parasitic capacitance.
- Power gating is implemented to all circuit blocks. The blocks not in operation are to be powered down completely. The unselected memory together with the computational cell will be disabled during memory update.

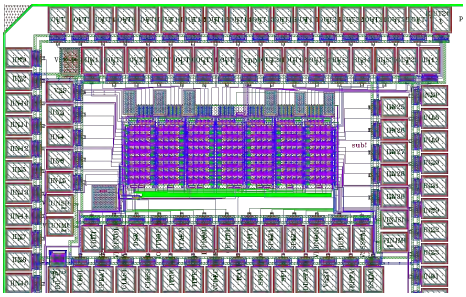


Figure 9: Layout of second prototype

- The biasing current for the floating-gate memory is reduced from 10 nA to 1 nA. Other bias currents are reduced as well.

- The V to I converter in the memory is changed from differential pair to a single-ended diff-pair. It yields similar sigmoid transfer function, but consumes about half of the power.

- A normalize-inverse circuit was designed to generate belief states (inputs to the higher level) from Mahanalobis distance outputs of the lower level.

- A continuous-time winner-take-all (WTA) is integrated with the normalize-inverse circuit to replace the time-domain LTA of last version.

- A sample-hold circuit for belief states (inputs of higher levels) is implemented to enable parallel operation of 3 layers. The hold capacitors are realized solely with parasitic capacitance to achieve good power efficiency.

- The input/output currents connecting off chip are scaled up to mitigate speed limitation.

- The layout has been made denser to reduce area and parasitic capacitance.

Task II: FPAA Investigations

Adaptive FPAA Testing

Task II work has included testing of a new FPAA device, which allows floating-gate adaption as elements in the CABs. This structure enables compiling of adaptive floating-gate circuits in a routing fabric made up of similar floating-gate circuits. We have an auto-zeroing floating-gate amplifier (AFGA) experimentally working, as well as a few related elements. We expect this system will be important to investigate some capabilities for the floating-gate learning part of the process.

Adaptive Circuit Implementations

We have tested an FPAA containing adaptive elements and shown it to be well-suited to the implementation of a variety of adaptive circuits. As an example, one adaptive circuit demonstrated in the adaptive FPAA is an adaptive differential pair, a variation on one of the most commonly used subcircuits in analog circuit design. An adaptive differential pair adjusts to input signals so as to return its output to a given equilibrium. By doing so, it effectively learns the mean of the input signal, a critical element in the clustering process. Figure 10 shows the schematic and measurements for an adaptive differential pair. Other adaptive blocks implemented thus far include auto-zeroing floating-gate amplifiers (AFGAs), correlating adaptive filters. In these tests, we have observed stable and competitive behaviors, consistent with custom IC designs developed at Caltech and GT.

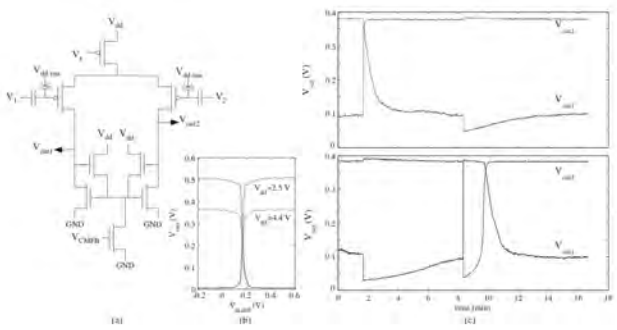


Figure 10: Schematic and measurement results for an adaptive differential pair.

Task III: Algorithm Modeling and Development

Progress on Task III has included modeling of the impact of various analog computational non-idealities and the development of a new architecture.

Error Sensitivity Analysis

The effects of transistor mismatch and noise on the clustering process were explored. In this section, several error sources will be described and their impact on a full-scale system will be presented.

Input Variation Each dimension of the input signal must be copied to each of the centroids. In current-mode signaling, this will be accomplished via a current mirror, and the input will be multiplied by a gain error. With voltage signaling, the same input can be physically shared between centroids without explicit copying, and this error source is avoided. This results in a distance calculation for centroid **i** in dimension **j** such that for a gain variation $\boldsymbol{d_{i,j} = f(\epsilon_{i,j} o_i, \mu_{i,j})}$. The error term $\boldsymbol{\epsilon_{i,j}}$ is different for every centroid and for every dimension within a centroid.
When modeled as a gain error, input variation error will have the effect of giving certain dimensions of the input more or less weight in the distance calculations and in the calculation of the belief state. The impact of this error will be dependent on the characteristics of the true data clusters. If they are well separated, no significant impact should be expected. If they are not, the system could characterize the input in a different manner than expected if each input dimension were given equal weight as demonstrated in Figure 11.

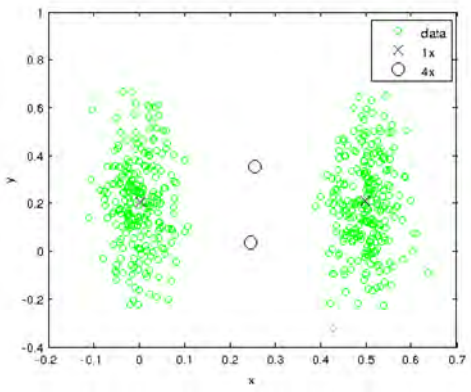


Figure 11: The effect of an input variation that is too large.

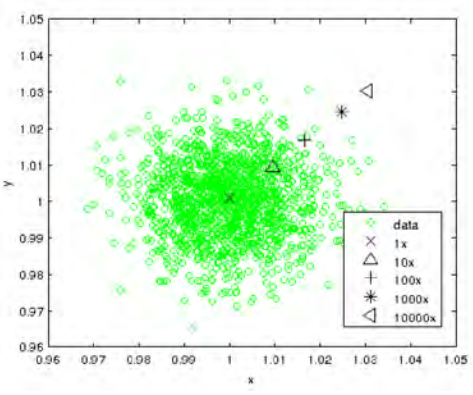


Figure 12: The effect of update asymmetry.

Update Asymmetry and Variation Mismatch in the update mechanisms may result in asymmetric updates,

such that increments to a given memory are of a different magnitude than decrements. Also due to variation in the update mechanism, the update rate may vary from one memory cell to another.
An asymmetric error will have some system level impact. During the transient stage of centroids moving towards their clusters, this error will simply cause the learning rate to be modified. During the steady state stage of centroids learning the mean and variance of their clusters, this will cause the mean to be offset within the cluster and the calculated variance to be larger. The expected offset is expected to be proportional to the mismatch between increments and decrements. Assuming the increment is larger than the decrement, if the number of increments and decrements are equal the centroid will move upwards. However, as the centroid moves upwards the number of decrements will increase and cause the centroid to reach equilibrium, $\boldsymbol{\lambda_{inc}}\,\mathbf{Pr}(\boldsymbol{x} > \boldsymbol{\mu}) = \boldsymbol{\lambda_{dec}}\boldsymbol{P}(\boldsymbol{x} < \boldsymbol{\mu})$.
A demonstration of this effect can be seen in Figure 12. Variation in the update rate between memory cells will have the effect of some centroids converging towards their equilibrium faster than others. This will have no effect on the equilibrium state of the centroids and no significant effect on the learning transient except in extreme cases.

Memory Adaptation Variation Each analog memory cell will tend to converge towards the inputs applied to it when it is updated. Input-referred offset or gain error here will cause the memory to converge to a value different than the actual mean of the cluster it is learning. This error can be represented by modifying the error term applied to the observation for the update equation with respect to that applied for the distance measurement.
If the memory adaptation variation is modeled as a gain error, it will have the effect of increasing or decreasing the learning rate. Much like the update variation error, it will have little to no effect except in extreme cases. However, if this error is modeled as an additive error, it will have the effect of shifting the learned centroid by the amount of the error. For small errors, this will have a small effect on the calculation of the belief state. When the error becomes larger, it can cause a centroid to walk far away from the data cluster it is supposed to represent and possibly towards another data cluster. The effect of such a case is demonstrated in Figure 13.

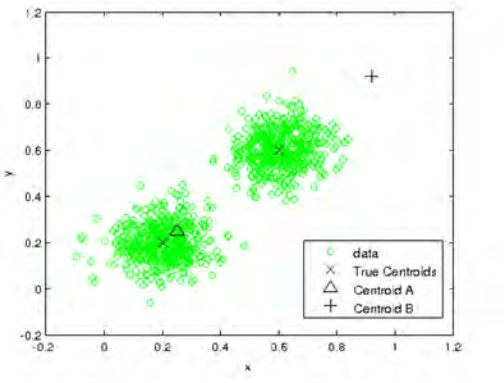


Figure 13: The effect of memory adaptation variation when modeled as a bias error.

Distance Error The distance measurement circuits may exhibit gain and/or offset errors. Variation within a given centroid could result in one dimension contributing disproportionately to the overall distance between an observation and an input. Circuitry accepting the individual one-dimensional inputs will contribute to error uncorrelated across dimension within a given centroid. Circuitry operating on the aggregated distance will contribute to an error that affects each dimension identically.
For reasonable error levels, distance error has the effect of giving one dimension more or less importance is the centroid selection process much like the case of the input variation.

Distance Comparison Error There may be input-referred offset in the distance comparison block, typically implemented as either a winner-take-all (WTA) circuit for similarity or a loser-take-all (LTA) for difference. The effect of this can be expected to be similar to a distance error that is identical for all dimensions of a given centroid, but that varies across centroids.

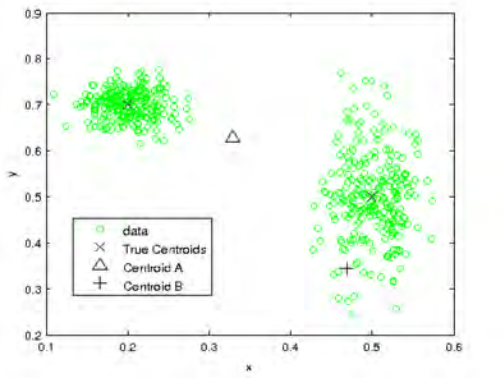


Figure 14: Centroid B has a distance comparison error causing it to appear artificially distant from any input.

On a system level, distance comparison error will have the effect of causing inaccurate belief state calculations and making a centroid appear artificially further from (or closer to) all inputs in the selection

algorithm. If this error is small, it will have no effect since the distance comparison is only used to select the centroid to update. If the error is larger, a centroid could become starved until this distance comparison error comes into equilibrium with the starvation trace. Once this equilibrium is reached, the centroid will still claim fewer input vectors than it should, since its starvation trace cannot remain small enough to claim input vectors without allowing other centroid to claim more input vectors. This effect is demonstrated in Figure 14.

Noise Each of the signals is an analog current or voltage and is thus subject to additive noise. The noise is typically Gaussian. It may be spectrally white such as shot noise and thermal noise, or pink concentrated at low frequencies, such as flicker noise.
Noise should have no effect on the calculated centroid means since it is a zero mean process. For clusters with a smaller variance than the noise level, the calculated variance will be similar to the noise level rather than similar to the true variance. If the true variances are less than the noise level for all centroids, then the beliefs will be calculated based on a distance measure that is approximately Euclidean distance than normalized Euclidean distance because of the inaccurate calculated variances. The beliefs will then have less information than they otherwise would, but the beliefs will still be meaningful because of the accurate mean values.

System-Level Impacts Given the inaccuracies discussed above, in this section we examine the impact of error sources on the algorithmic performance. This is accomplished by using computational models of the inaccuracies present in analog computation extracted from transistor level simulations of the circuit. First, a metric for algorithmic performance must be defined. Since the common method for using DeSTIN in pattern recognition tasks is to extract the beliefs as features, performance is defined as the mean absolute error (MAE) between what the ideal belief values should be and those calculated considering errors in the system. It is important to note that this means performance is not directly tied to the numerical accuracy of the calculated centroid means and variances, which are being calculated in a space altered by the error sources. In the remainder of this section, we first explore the effects of the analog error sources on a synthetic dataset in order to demonstrate the effect on calculated belief states when the true centroids are known. Then, we present a larger scale classification problem using the MNIST dataset in which classification performance is the best metric to evaluate the success of the system.

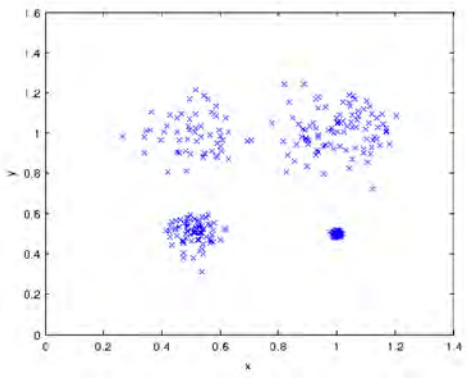


Figure 15: Data used for belief MAE calculations.

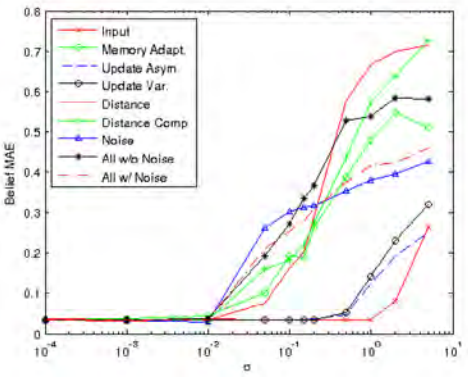


Figure 16: Belief MAE when all errors are modeled as bias errors.

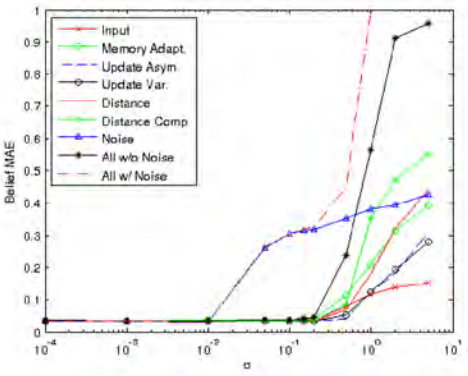


Figure 17: Belief MAE when all errors are modeled as gain errors except noise.

To demonstrate the effect of the various errors and mismatches, a simple clustering problem is considered. The data shown in Figure 15 is clustered using a single DeSTIN node with varying levels of error and noise. The noise is always additive Gaussian noise, while the remaining inaccuracies are modeled both as gain errors and additive errors. Noise is added to select signals in the system in the same manner as additive error. The use of currents to represent variables generally leads to gain errors, while voltage-based signals lead to offset errors. We modeled the cases of full current-mode signaling and full voltage-mode signaling to explore the different effects. The impact of gain and offset errors on a system using mixed-mode signaling can be expected to fall between these two cases.

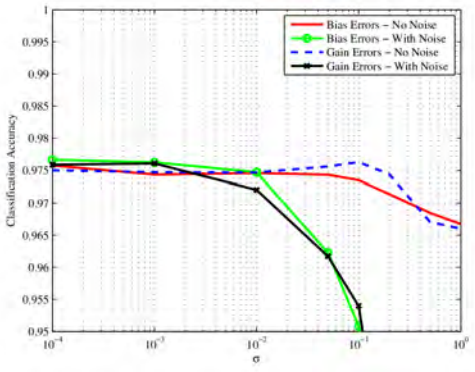


Figure 18: Classification results on the MNIST dataset with varying levels of error/noise.

The resulting error in the belief is calculated as the mean absolute error between the ideal belief vector and that obtained using analog computation. Figures 16 and 17 illustrate the effect of the errors discussed in this section on a single node's belief values. As can be observed, none of the errors introduce any notable degradation below a standard deviation value of 10^{-2}. When modeling all errors as gain errors, additive noise in the system has a much larger impact than even the rest of the errors combined. Thus, it is demonstrated that the inconsistency caused by noise is much more harmful than the consistent gain and bias errors. The most destructive gain errors are the distance comparison, distance, and memory adaptation variation errors. The update asymmetry, input, and update variation errors have much less impact.

When all errors are modeled as bias errors, additive noise has much the same effect as the distance comparison, distance, and memory adaptation variation errors. The update asymmetry, input, and update variation errors still have much less impact.

It is important to remember, however, that "correct" clustering as defined by the metric used for these clustering datasets is not necessary for the DeSTIN hierarchy to produce meaningful beliefs. To test this ability, a full DeSTIN hierarchy is utilized to form features for a standard multi-layer perceptron neural network classifier. The DeSTIN hierarchy used contains three layers. The top, middle, and bottom layers contain 1, 4, and 16 nodes respectively, with the nodes in each layer using 25, 18, and 25 centroids. The belief states are then sampled from three movements over the image. These beliefs are then provided to the classifier. Thus, we are testing the ability of our system which contains noise and errors to form a model of the regularities it observes and to produce features for a MLP classifier. The classifier used is a feed forward neural network with two hidden layers with 128 neurons in the first hidden layer and 64 neurons the second hidden layer. The dataset used is the MNIST hand written digits dataset. Results are shown in Figure 18, where the robustness of the DeSTIN architecture to the gain errors, additive errors, and additive noise introduced by the analog computation models is illustrated. In particular, the errors have little to no effect until the standard deviation of the errors becomes larger than the operating range (0 to 1). High levels of noise result in inconsistent beliefs that hold less information for the classification module, as one would expect, but the system can be seen to be robust to significant levels of noise.

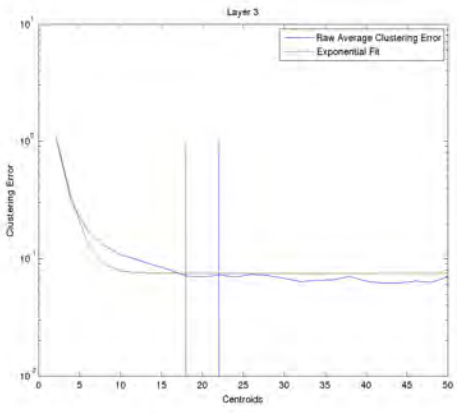


Figure 19: Illustration of the clustering error as a function of the number of centroids

The results from the clustering performance tests and the MNIST classification test allow some important conclusions to be drawn. A significant amount of error and noise can be introduced to the DeSTIN architecture and its clustering algorithm without having a destructive effect upon performance. It is particularly

noteworthy that noise is the most harmful error source by a significant margin. This is intuitively reasonable, as noise represents a dynamic error source to which the learning system cannot adapt. In contrast the other error sources distort the input space in a static way, leaving relationships in the underlying data intact. Even noise does not significantly degrade MNIST performance for normalized levels below 10^{-2}, corresponding to a signal-noise ratio (SNR) of 40~dB.

Automated Topology Optimization

We created an automated method of determining the topology (number of centroids per layer) of a DeSTIN network. The method is based on observing when adding more centroids no longer significantly decreases the clustering error. For each layer, we increase the number of centroids until the rate at which the clustering error decreases with respect to the number of centroids falls below a threshold. This is set as the number of centroids for that layer, and then the layer above completes the same process. The results are shown in Figure 19. This has resulted in a slightly different topology than we had been using and a small improvement in performance. This method will also provide guidance in the design of the IC prototypes in selecting an appropriate number of centroids for the chosen test problem.

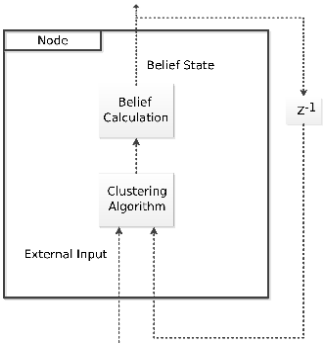


Figure 20: Recurrent clustering.

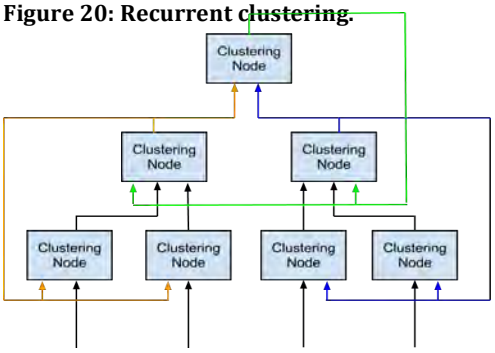


Figure 21: New hierarchical structure.

Behavioral Circuit Models

We built Verilog-A behavioral models of the sub-circuits of the clustering system described under Task I. This has allowed us to explicitly vary non-ideal effects (e.g. offsets, etc.) to see the effect on clustering performance. It additionally enables mixed-mode simulations, where some sub-circuits are simulated at the transistor level while others are simultaneously simulated with much more efficient (but potentially less accurate) behavioral models. These behavioral models offer a useful middle ground between complete transistor-level models

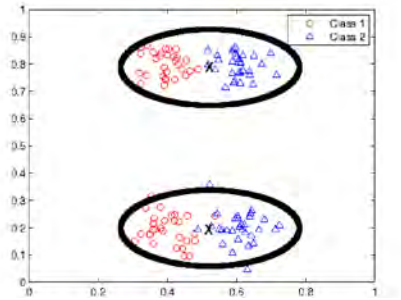


Figure 22: Clustering without using class inf

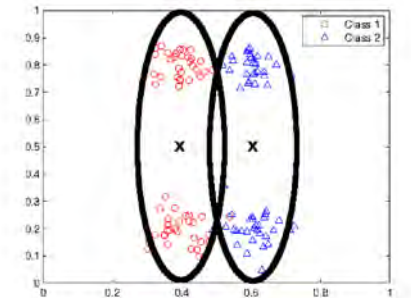


Figure 23: Clustering using class information.

Simplified Node Architecture

We have developed a simplified architecture for the node populating DeSTIN, which embeds a feedback-mechanism to inherently capture temporal dependencies in a more natural manner than that of the original scheme. In order to more easily map the DeSTIN architecture to an analog implementation, the mechanism used by the original DeSTIN architecture to pass information top-down, reflected by the construct $\mathbf{Pr}(\boldsymbol{s_t} | \boldsymbol{s_{t-1}}, \boldsymbol{a_{t-1}})$, needed to be revised. The philosophical approach taken was to integrate the feedback/recurrence mechanism as an inherent part of the clustering process. In previous work, the temporal regularities $\mathbf{Pr}(\boldsymbol{s_t} | \boldsymbol{s_{t-1}}, \boldsymbol{a_{t-1}})$, were captured by maintaining a table populated with the likelihoods of transitioning between states or through a function approximation method that attempted to predict the next

state given the current state. Though keeping a table of transition probabilities seems simple enough, the manner in which it was being used necessitated that an array be kept for every movement made across the image and for every possible belief state provided by the parent node. This results in a table that has a memory requirement for a single node of $\boldsymbol{M_{tab} = K^2AL}$, where **K** is the number of centroids for the node, **L** is the number of movements, and **A** is the number of belief states from the parent. In addition to the large memory requirement, using this table mandated an additional set of operations outside of the core node functionalities of clustering and calculating $\mathbf{Pr}\ (\boldsymbol{o}|\boldsymbol{s_t})$. The other previously used mechanism to estimate $\mathbf{Pr}(\boldsymbol{s_t}|\ \boldsymbol{s_{t-1}}, \boldsymbol{a_{t-1}})$ is function approximation. While this method has a more modest memory footprint, it requires an extensive set of operations outside the core clustering functionality of the node. These additional operations make it incredibly difficult to map the DeSTIN architecture to an analog implementation. For this reason, it is desired to couple the learning of temporal regularities and parent advice more closely with the clustering mechanism.

To address this problem, recurrent clustering is proposed as illustrated in Figure 20. Recurrent clustering takes as input the external input augmented by the node's previous time step belief. This allows the clustering to form beliefs that are based on both spatial and temporal attributes. Consequently, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma^2}$ have dimensionality $\boldsymbol{K(N+K)}$, where **K** is the number of centroids and **N** is the number of input dimensions. It is important to note that in addition to allowing the clustering mechanism to capture temporal dependencies, this method facilitates the formation of centroids that represent relationships between the spatial and temporal features of the data. During the clustering process, the centroids will converge to values that represent spatial and temporal regularities in the data. Previously, the clustering algorithm could only observe the input vector and characterize its similarity to other input vectors. In this revised formulation, the clustering results in belief states that represent information about transitions between belief states or, more generally, about a sequence of transitions between belief states, since each belief depends on the preceding belief. The revised architecture is outlined in Figure 21. We've been evaluating the revised architecture on standard benchmarks, such as MNIST, with encouraging results essentially identical to the ones obtained using the previous node design.

Task-Guided Feature Extraction

We have been refining and evaluating a system which augments DeSTIN (deep learning based feature extractor) with labeled data in order to form features that aid in the classification task. As anticipated, such a mechanism helps improve performance by guiding DeSTIN to form features across the hierarchy which are not just driven by regularities in the observations but also by the goals of the application that uses the features. While unsupervised clustering methods strive to capture regularities in a dataset, they do not necessarily capture the most relevant regularities in order to classify the samples. In order to better represent the differences between classes, the class labels can be used to help ensure that the clustering model captures regularities that help discriminate the classes. By appending the label, **y**, to the input vector, **o**, a new input vector, $\widehat{\boldsymbol{o}} = [\boldsymbol{o}\ \boldsymbol{y}]$, is formed that allows improved centroids to be formed. In some cases, the original input vector and the label can have very different scales or distributions and cause an imbalance in the importance attributed to the original input vector, **o**, and the label, **y**. In this case, a scaling factor, $\boldsymbol{\beta}$, can be used to modify the range of the output variable, $\widehat{\boldsymbol{o}} = [\boldsymbol{o}\ \boldsymbol{\beta y}]$. The centroids, $\widehat{\boldsymbol{\mu}} = [\boldsymbol{\mu}\ \boldsymbol{\mu_y}]$, that are learned during this supervised training can then be projected onto the space consisting of only the original input vectors for testing. We observe notable improvement in Figure 14 due to the feedback and feel confident it can be realized in hardware with ease.

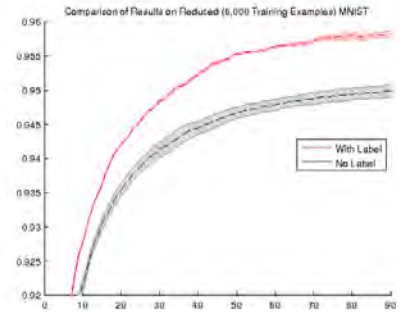


Figure 14: Classification results using improved clustering.